LES FICHES CUISINE D'ONC POSTSCRIPT : 2. UN ÉMULATEUR DE TABLE TRAÇANTE

Bruno Borghi

Cahiers GUTenberg, nº 3 (1989), p. 66-68.

<http://cahiers.gutenberg.eu.org/fitem?id=CG_1989___3_66_0>

Les fiches cuisine d'Onc' PostScript
Fiche N° 2 : un émulateur de table traçante

Bruno Borghi

METASOFT, 13 rue Duhamel, 35000 Rennes

1. Résumé des épisodes précédents

La fiche N° 1 était consacrée à la réalisation du programme *"Pen Plotter"*. Ce programme offre les fonctionnalités d'une table traçante au moyen de 4 procédures rassemblées dans un préambule. Je n'en rappelle ici que de succinctes spécifications[1] :

- *bop* -	Beginning Of Page
- *eop* -	End Of Page
x y *u* -	pen Up, moving to (x,y)
x y *d* -	pen Down, moving to (x,y)

On induit ainsi un format de description de page, que j'appellerai dans la suite *PP* pour *Pen Plotter.* Voici un exemple de fichier dans ce format.

```
bop
100 120 u
54 25 d
66 27 d
100 120 d
eop
```

2. Tout se complique

Dans la vraie vie, les programmes du commerce ne génèrent pas des fichiers au format PP. Ils utilisent divers formats « standards » pour table traçante, qui se ressemblent tous.

Considérons le format « standard » OPGL *(Onc' PostScript Graphic Language)*, qui définit 2 opérateurs :

U x y	pen Up, moving to (x,y)
D x y	pen Down, moving to (x,y)

En OPGL, on ne peut décrire qu'une page à la fois ; chaque ligne de la description ne comporte qu'un opérateur. Un fichier dans ce format a l'allure générale suivante :

```
U 100 120
D 54 25
D 66 27
D 100 120
```

OPGL est très proche de PP, mais il adopte une syntaxe radicalement différente.

- Le format PP est *postfixé* comme l'est PostScript : l'opérateur apparaît *après* les opérandes.
- Par contre, le format OPGL est *préfixé* : l'opérateur apparaît *avant* les opérandes.

Comment transformer notre imprimante PostScript en imprimante OPGL?

3. Fabriquons notre imprimante OPGL

Pour pouvoir imprimer des fichiers OPGL, nous allons écrire en PostScript un inter-

[1] La fiche N° 1 est parue dans le N° 1 des Cahiers de Gutenberg. Tout est expliqué dedans. Il y a même une cinquième procédure.

préteur du format OPGL. Le schéma de fonctionnement d'un tel interpréteur est le suivant :

- le fichier de description du tracé est lu ligne par ligne ;
- pour chaque ligne, l'instruction OPGL est décodée et exécutée.

La procédure principale est une boucle de lecture, que nous allons réaliser avec les opérateurs `currentfile` et `readline`.

```
% reading loop
   /line-buffer 80 string def

   /main-loop {
      bop
      {
         currentfile line-buffer
         readline
         exch parse-line
         not {exit} if
      } loop
      eop
   } def
```

`readline` lit une ligne dans le fichier désigné par `currentfile`, c'est à dire, dans le cas général, sur l'entrée standard de l'imprimante. *Ainsi, les lignes au format OPGL sont lues directement par la procédure* `main-loop` *plutôt que par l'interpréteur PostScript.*

La boucle de lecture se termine quand une indication de fin de fichier est rencontrée. Vous noterez la structure de cette boucle : la dernière ligne est interprétée correctement même si le fichier de description ne se termine pas par le caractère de fin de ligne.

D'autre part, les spécifications du format PP nous conduisent à encadrer la boucle de lecture par les procédures `bop` et `eop`.

Maintenant, il nous faut définir la procédure `parse-line` qui interprète chaque ligne. L'opérateur PostScript `token` sert à extraire la première entité syntaxique d'une chaîne de caractères, selon la syntaxe de PostScript. Le découpage en entités syntaxiques de PostScript convient pour OPGL, même si les syntaxes sont différentes. Nous pouvons donc utiliser `token` pour faire ce travail.

```
% execute a (possibly empty) line
   /parse-line {% string parseline
      token {
         /op exch def
         mark
         token {{exch token}
         {exit} ifelse} loop
         op
         cleartomark
      } if
   } def
```

Le premier appel de `token` extrait l'opérateur OPGL. Si la ligne est vide, la procédure est terminée. Si la ligne n'est pas vide, on «poussé» sur la pile opérandes tous les arguments trouvés à la suite de l'opérateur. On retrouve l'ordre postfixé habituel de PostScript : l'opérateur peut être alors exécuté.

La structure en boucle adoptée pour l'extraction des arguments permet de définir une procédure générale, valable quelque soit le nombre d'arguments de chaque opérateur. On pourra ainsi faire évoluer facilement l'interpréteur si de nouveaux opérateurs sont ajoutés au format OPGL.

Une précaution supplémentaire : l'emploi de `mark` et `cleartomark` garantit qu'il ne restera sur la pile aucun argument indésirable pour la suite.

Il nous reste enfin à définir les opérateurs OPGL, ce qui ne pose pas de problème particulier.

```
% OPGL operators
  /U /u load def
  /D /d load def
```

4. Et maintenant, essayons

Voilà, notre imprimante PostScript est devenue un émulateur de table traçante OPGL. En voici le mode d'emploi :

1. concaténer les définitions de *Pen Plotter*, les définitions de l'interpréteur OPGL, l'appel à `main-loop` et les données au format OPGL ;
2. envoyer le fichier ainsi obtenu sur l'imprimante.

Le fichier final a l'allure suivante[2] :

```
%%%% Pen Plotter
  /bop [...] def
  /eop [...] def
  /u [...] def
  /d [...] def

%%%% OPGL emulator
  /main-loop [...] def
  /parse-line [...] def
  /U [...] def
  /D [...] def

%%%% loop and data
main-loop
U 100 120
D 54 25
D 66 27
D 100 120
```

5. Conclusion

Dans le cas étudié, nous avons eu la chance de pouvoir utiliser `token`. Si la syntaxe du format à interpréter est trop différente de celle de PostScript, `token` ne convient plus et il faut écrire une procédure spécifique (`mytoken`). Les opérateurs de manipulation de chaînes de caractères comme `search`, `anchorsearch`, `getinterval`, permettent de s'en tirer avec élégance.

On peut donc écrire un véritable interpréteur en PostScript. La structure générale de notre émulateur OPGL est valable pour la plupart des émulateurs de systèmes d'impression, que ce soit une table traçante, une imprimante ligne ou une imprimante graphique.

[2] Le lecteur remplacera, bien sûr, les ellipses par leur valeur.